AFRL-OSR-VA-TR-2014-0066

A LANGUAGE-BASED APPROACH TO WIRELESS SENSOR NETWORK SECURITY

Christian Skalka
UNIVERSITY OF VERMONT & STATE AGRICULTURAL COLLEGE

03/06/2014
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTC
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
2/24/2014

2. REPORT TYPE
Final

3. DATES COVERED *(From - To)*
4/1/09-11/31/2013

4. TITLE AND SUBTITLE
A LANGUAGE-BASED APPROACH TO WIRELESS SENSOR NETWORK SECURITY

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-09-1-0083

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Christian Skalka

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Vermont, Burlington VT 05405

8. PERFORMING ORGANIZATION REPORT NUMBER
22612

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research, Arlington VA

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In this project we investigated software architectures to support secure and reliable embedded systems application programming. In particular, we were focused on Wireless Sensor Network (WSN) applications developed the TinyOS programming environment. A major contribution was an extension of the nesC programming language, called SpartanRPC, that provides a secure RPC abstraction and an expressive and fine-grained security policy language. SpartanRPC allows multiple networks in distinct security domains to interact without resorting to other hardware intermediaries. Another major contribution was a staged programming language, called Scalaness/nesT, that allows nesC programs to be automatically generated by Scala programs running on a high-powered "hub" machine, and subsequently deployed over-the-air to WSN componet nodes. This framework allows more efficient, reliable, type-safe orchestration of WSN systems. Another significant contribution of our project was a new mechanism for embedding watermarks in WSN data to support fair-use policies for information sharing.

15. SUBJECT TERMS
Embedded Systems, Software Security, Programming Languages, Formal Methods.

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	UU	1	Christian Skalka
					19b. TELEPHONE NUMBER *(include area code)* 802 656 1920

FA9550-09-1-0083 Final Performance Report: Executive Summary

Objectives

The overall goal of the project was to develop new foundations and architectures for secure and reliable wireless sensor network (WSN) applications. Main subgoals included the following:

1. To develop a new middleware system supporting secure RPC communications in WSNs.
2. To develop new foundations and practical language architectures for type-safe *program specialization*, aka *staged programming*, in WSNs.
3. To develop a software framework for reliable association of provenance information with datasets.

Accomplishments

In this Section we detail accomplishments with respect to each of the objectives mentioned above.

Objective 1

Our work on secure RPC in WSNs focused on tools for the popular TinyOS programming environment. A major contribution was an extension of the nesC programming language, called SpartanRPC, that provides a secure RPC abstraction and an expressive and fine-grained security policy language. SpartanRPC allows multiple networks in distinct security domains to interact without resorting to other hardware intermediaries.

Our system includes an asynchronous RPC abstraction based on nesC wiring constructs. The benefit of this approach is that experienced nesC programmers will readily understand the usage of novel SpartanRPC constructs. Furthermore, programmers can specify dynamic endpoints of wirings, to accommodate changes in network configurations.

RPC services are secured with authorization policies mediating access. These policies are written in an expressive language that supports distributed, decentralized specification and maintenance. Individual security domains are able to manage their own policy specifications. For network communications, our underlying protocols use public keys to support an *open-world* security model, where security domains do not need to share secrets *a priori*. Rather, authorization is established via signed credentials communicated over the air. Because public key signature verification is very costly in WSNs, our protocols also incorporate symmetric session key negotiations for efficient communications with authorized actors.

Performance overhead of this system was measured using a variety of metrics. The majority of computational costs are incurred during initial authorization periods between network nodes, when credentials are verified and session keys are computed. Following this short transient state (90 seconds to several minutes depending on network densities), normal network communications proceed with relatively little overhead. General impacts are summarized in Fig. 1. To demonstrate memory overhead, we compare RAM and ROM bytes consumed in a simple client-serve test harness, implemented in "baseline" fashion with no security, and implemented using the SpartanRPC framework in a secure fashion. We also illustrate impact on maximum messaging rates, by comparing such rates in the baseline implementation with both insecure and secure RPC versions. The latter data shows that the most significant messaging overhead incurred by our system is from the security features, not the RPC abstraction provided in SpartanRPC.

Test Program	*RAM Bytes*	*ROM Bytes*
Baseline Client	349	10982
Baseline Server	283	10490
SpartanRPC Client	2222	23108
SpartanRPC Server	2126	23394

Test	*messages/sec*	*% Reduction*
Baseline	128	–
RPC	119	7.0
Secure RPC	87	32.0

Figure 1: SpartanRPC Memory Overhead (L) and Impact on Messaging (R)

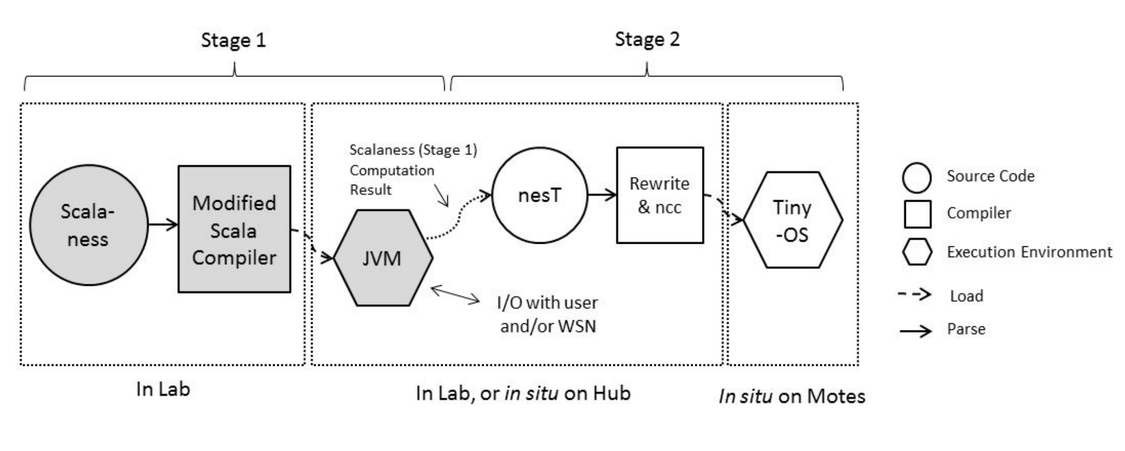

Figure 2: Scalaness/nesT Compilation and Execution Model

Objective 2

Due to resource constraints in WSNs, application efficiency is extremely important. At the same time, complex, distributed algorithms are typical application components. We proposed *program specialization*, aka *staged programming*, as a technique to address both these issues. In a staged programming language, code is treated as a datatype, and programs can be dynamically generated and subsequently executed. We envisioned that a staged programming language could be executed on a high-powered WSN network "hub" device, that could dynamically generate specialized, efficient WSN node programs based on conditions. A particular novelty of our approach was a focus on type safety, to ensure that type safe hub programs are guaranteed to generate type safe node programs. We call this *cross-stage type safety* This is especially important in case hubs are remote and not accessible by operators who could bug-fix type errors in generated code.

Although a variety of staged programming languages exist, they have not been designed with cross-stage type safety, or in an execution model that reflects WSN hardware architecture. Thus, a number of foundational issues existed. We addressed these issues in the $\langle ML \rangle$ language, which combined a core-ML language with staging features. Our main result was a formal proof of cross-stage type safety for $\langle ML \rangle$, though we resolved a variety of practical language design issues.

On this foundation, we next developed a practical staged language for developing real WSN applications. This language, called Scalaness/nesT, extends Scala with staging features for executing programs on hubs, that generate type-safe nesC programs for subsequent deployment to WSN nodes. The language compilation and execution model is described in Fig. 2. Of particular note here is the fact that cross-stage type safety of Scalaness source code ensures that compiled bytecode can be deployed to, and run on, hubs without fear of generating ill-typed specialized WSN node programs.

To explore and demonstrate practical applications of Scalaness/nesT, especially in a security setting, we re-implemented the SpartanRPC system in a staged style. In particular, we offloaded credential verification and symmetric key computations to a Scalaness program on a hub, that generates specialized node

		Unsecured	Unstaged	Staged	Savings
Server:	ROM	36254	48616	36596	25%
	RAM	2868	5417	3038	44%
Client:	ROM	24316	35834	24436	32%
	RAM	2274	4771	2402	50%

Figure 3: Comparison of Staged and Unstaged Version of SpartanRPC-secured Application

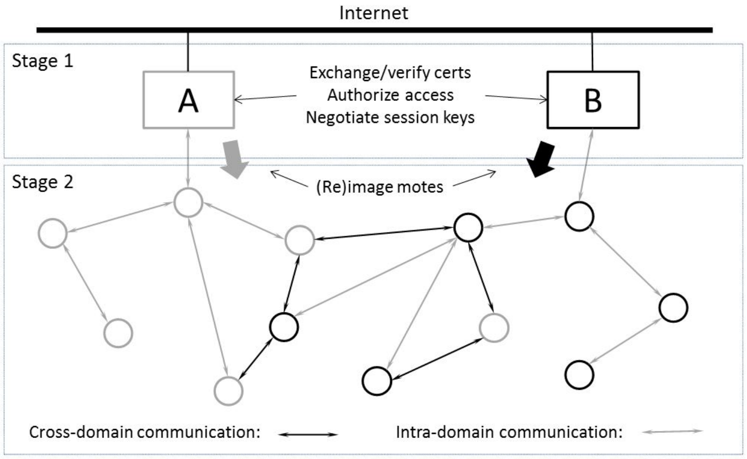

Figure 4: Staging Authorization and Authorized Access in a Multi-Domain WSN.

programs with embedded session keys for secured network communications. This system is illustrated in Fig. 4. A staged approach to SpartanRPC communications has a significant effect on RAM and ROM usage, as summarized in Fig. 3. We compared three software versions of a client-server application: one with no security mechanisms in place, one with unstaged SpartanRPC protocols in place, and one generated by Scalaness evaluation in our staged version of the SpartanRPC protocol. The "Savings" in this figure are the percent reduction from unstaged to staged secure implementation, and these numbers show the potential for saving both RAM and ROM space is quite significant using staging. This is especially important, since it changes the delicate balance between the security and language abstraction benefits of SpartanRPC, and associated implementation overhead.

Objective 3

Wireless Sensor Networks typically produce time-series data that is frequently intended for the public domain. But even if data is shared publicly, it is still important to maintain metadata, especially provenance, for proper understanding and attribution of data sources. Most metadata schemes for environmental data are based on imposed structure and annotations, e.g. XML formats. However, this scheme is brittle, due to typical practices of domain scientists, who are mainly interested in data, who perform analysis and spreadsheets, and who are likely to "throw away the (metadata) wrapper".

Thus, we proposed a scheme for embedding identifiers directly in time series data, and associating those identifiers with web-accessible provenance information. This scheme is called *self-identifying data*. Self-identifying data leverages noise in sensor readings, in a manner similar to existing watermarking techniques. To anticipate manipulations of data common in scientific practice, the scheme is robust to data sampling, reordering, and truncation. Although the scheme is not secure, in the sense that it can be easily subverted by a malicious actor, it is rather intended to support "Fair-Use" data sharing policies between

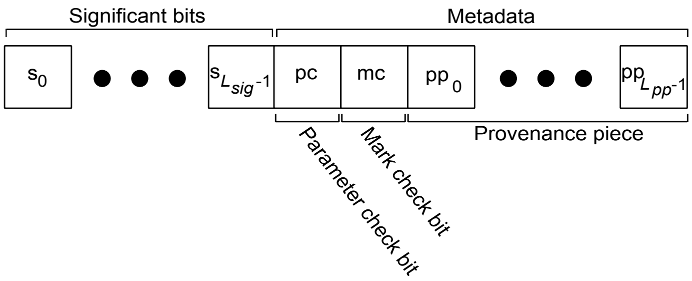

Figure 5: Anatomy of an Annotated Datapoint

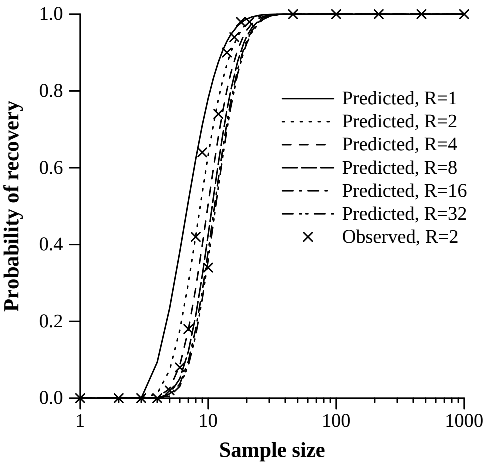

Figure 6: Predicted and observed probability of recovering provenance mark.

well-intentioned actors. Support for Fair Use policies is especially important, since concerns about proper attribution is one of the most common barriers to data sharing.

Our technique is illustrated in Fig. 5. In each datapoint, some range of least significant bits is reserved for metadata embedding– the size of this range is dependent on the accuracy and precision of sensor readings. More significant bits are left untouched. Bits reserved for metadata are used to encode a fragment of the embedded provenance identifiers, while the parameter and mark check bits are used to determine fragment ordering and validity for reconstructing complete marks. Because each marked datapoint contains only a fragment of the complete provenance identifier, identifier reconstruction requires multiple datapoints.

In Fig. 6, we show both a combinatorial prediction and empirical observations of the probability of recovering a provenance identifier given a particular data sample size. Empirical observations for several datasets are shown. In summary, our results illustrate that a sample of 30-40 distinct datapoints are adequate for near 100% probability of provenance identifier recovery. In addition to this analysis, we have illustrated practical application of our scheme in a publicly available dataset generated by environmental monitoring networks. This dataset is paired with a web tool that automatically extracts provenance identifiers from datasets, and redirects to webpages with provenance information.

Personnel Supported

The following graduate students and postdoctoral researchers were supported on this grant in the manner noted.

- Evgeny Makarov, Postdoctoral Researcher (6 months support).
- Simone Willett, MS, graduated May 2012 (full support for entire degree program).
- Michael Watson, MS, expected graduation May 2014 (full support for entire degree program)..
- Peter Chapin, PhD, graduated December 2013 (travel support only).
- Sepehr Amir-Mohhamadian, PhD (11 months support).

Related Publications, Including Theses

Note: Publications listed report on work either directly related to, or supportive of, grant funded research. In particular, we list publications reporting on application and empirical study settings for developed software systems.

- Stephen Chong, Christian Skalka, and Jeffrey A. Vaughan. *Self-Identifying Sensor Data*. Submitted for review to the ACM Journal of Data and Information Quality, 2014.
- Peter Chapin and Christian Skalka. *SpartanRPC: Remote Procedure Call Authorization in Wireless Sensor Networks*. Invited for Resubmission with Major Revisions to ACM Transactions on Information and Systems Security, 2014.
- Peter Chapin. *Trust Management in Distributed Resource Constrained Embedded Systems*. PhD Thesis, Department of Computer Science, University of Vermont, Fall 2013.
- Michael Watson. *Type Checking Implementation in Scalaness/nesT*. MS Thesis, Department of Computer Science, University of Vermont, Spring 2014 (expected defense date March 25, 2014).
- Peter Chapin, Christian Skalka, Scott Smith, and Michael Watson. *Scalaness/nesT: Type Specialized Staged Programming for Sensor Networks*. In ACM Generic Programming: Concepts and Experiences (GPCE), 2013.
- Christian Skalka and Jeff Frolik. *Snowcloud: A Complete System for Snow Hydrology Research*. In ACM Workshop on Real-World Wireless Sensor Networks (RealWSN), 2013.
- Yu David Liu, Christian Skalka, and Scott Smith. *Type-Specialized Staged Programming with Process Separation*. Journal of Higher Order and Symbolic Computation, 24(4):341-385, 2012.
- C. David Moeser, Mark Walker, Christian Skalka, and Jeff Frolik. *Application of a wireless sensor network for distributed snow water equivalence estimation*. In Western Snow Conference, 2011.
- Peter Chapin and Christian Skalka. *SpartanRPC: WSN Middleware for Cooperating Domains*. In IEEE Conference on Mobile and Ad-Hoc Sensor Systems, 2010.
- Stephen Chong, Christian Skalka, and Jeffrey A. Vaughan. *Self-Identifying Sensor Data*. In ACM Conference on Information Processing in Sensor Networks (IPSN), 2010.
- Yu David Liu, Christian Skalka, and Scott Smith. *Type-Specialized Staged Programming with Process Separation*. In Workshop on Generic Programming (WGP09), Edinburgh, Scotland, 2009. Note: this version contains minor revisions to the article published in WGP09 Proceedings.
- Evgeny Makarov and Christian Skalka. *Formalized Proof of Type Safety of Hoare Type Theory*. Technical Report CS-09-01, The University of Vermont, 2009.

Meeting Participations and Presentations

Note: Presentations listed publicized work either directly related to, or supportive of, grant funded research. In particular, we list presentations on application and empirical study settings for developed software systems.

- Peter Chapin, Christian Skalka, Scott Smith, and Michael Watson. *Scalaness/nesT: Type Specialized Staged Programming for Sensor Networks*. Presented at the ACM Conference on Generic Programming: Concepts and Experiences (GPCE), 2013.
- Christian Skalka and Jeff Frolik. *Snowcloud: A Complete System for Snow Hydrology Research*. Presented at the ACM Workshop on Real-World Wireless Sensor Networks (RealWSN), 2013.
- Christian Skalka, Joshua Bongard, Jeffrey Frolik, and Ian Brown. *The Snowcloud System: Architecture and Algorithms for Snow Hydrology Studies*. Poster presented at the American Geophysical Union (AGU) Fall Meeting, 2013.
- Peter Chapin, Christian Skalka, Scott Smith, and Michael Watson. *Scalaness/nesT: Type Specialized Staged Programming for Sensor Networks*. Presented at the Northeastern University Programming Languages Seminar, May 2013.
- Stephen Chong, Christian Skalka, and Jeffrey A. Vaughan. *Self-Identifying Sensor Data*. Presented at the 2012 Dagstuhl Seminar on Provenance in Software Systems.
- C. David Moeser, Mark Walker, Christian Skalka, and Jeff Frolik. *Application of a wireless sensor network for distributed snow water equivalence estimation*. Presented at the Western Snow Conference, 2011.
- Peter Chapin and Christian Skalka. *SpartanRPC: WSN Middleware for Cooperating Domains*. Presented at the IEEE Conference on Mobile and Ad-Hoc Sensor Systems, 2010.
- Stephen Chong, Christian Skalka, and Jeffrey A. Vaughan. *Self-Identifying Sensor Data*. Presented at the ACM Conference on Information Processing in Sensor Networks (IPSN), 2010.
- Yu David Liu, Christian Skalka, and Scott Smith. *Type-Specialized Staged Programming with Process Separation*. Presented at the ACM Workshop on Generic Programming (WGP09), Edinburgh, Scotland, 2009.

Patent Disclosure

The following patent awarded in 2013 is relevant to real WSN environmental monitoring systems developed by us at UVM, that served as practical testbeds for a variety of research performed in this project.

- A distributive, non-destructive real-time approach to snowpack monitoring, w/ J. Frolik, University of Vermont, US Patent No. 8,552,396 (Issued: October 8, 2013).